УДК 519.854

С.В. Листровой [1], д-р техн. наук,
Е.С. Листровая [2], канд. техн. наук, **М.С. Курцев** [1], аспирант
[1] Украинский государственный университет железнодорожного транспорта
(Украина, 61050, Харьков, пл. Фейрбаха, 7,
тел. (050) 3029912; e-mail: om1@yandex.ru; kurtsev_m@ukr.net),
[2] Национальный аэрокосмический университет им. Н.Е. Жуковского,
(Украина, 61070, Харьков, ул. Чкалова, 17,
e-mail: listravkina@gmail.com)

Ранговый подход к решению задач линейного и нелинейного булевого программирования для планирования и управления в распределенных вычислительных системах

Показана эффективность рангового подхода к решению произвольных задач булевого программирования. Описаны процедуры, позволяющие решать задачи линейного и нелинейного программирования с использованием алгоритмов полиномиальной сложности с небольшой погрешностью и произвольными нелинейностями как в функционале, так и в ограничениях. Приведены результаты экспериментального исследования погрешности разработанных алгоритмов и их временной сложности.

Показано ефективність рангового підходу до вирішення довільних задач булевого програмування. Описано процедури, які дозволяють розв'язувати задачі лінійного і нелінійного програмування з використанням алгоритмів поліноміальної складності з невеликою похибкою та довільними нелінійностями як в функціоналі, так і в обмеженнях. Наведено результати експериментального дослідження похибки розроблених алгоритмів та їх складності.

К л ю ч е в ы е с л о в а: дискретное программирование, ранговый подход, планирование, распределенная вычислительная система.

Задачи булевого линейного и нелинейного программирования являются моделями широкого класса прикладных задач в теории построения сложных систем. При этом задачи булевого линейного программирования относятся к классу *NP*-полных трудно решаемых задач [1], а эффективные методы решения задач нелинейного булевого программирования с произвольными нелинейностями практически отсутствуют. В настоящее время для каждого типа задач разрабатывается собственный метод решения.

© С.В. Листровой, Е.С. Листровая, М.С. Курцев, 2017

Представляется актуальным выработать единый подход, обеспечивающий решение данного класса задач с требуемой оперативностью и точностью. Последнее обусловлено тем, что в настоящее время получили широкое распространение распределенные вычисления, ставшие в масштабах глобальной сети одним из ключевых направлений развития сетей, которое расширяет наше представление о способах использования вычислительных ресурсов (ВР) и сети в целом. Об этом свидетельствует появление Grid технологий и облачных вычислений, в которых сеть рассматривается как единый ВР.

Основные направления, откуда берут начало Grid технологии, — это параллельные и распределенные вычисления. Именно синтез знаний из этих двух областей и их применение необходимы для реализации концепции распределенных вычислительных сред. Распределенные вычисления в масштабах глобальной сети становятся одним из ключевых направлений развития сетей [2, 3].

Предлагаемый подход позволяет решать задачи булевого линейного и нелинейного программирования с произвольными нелинейностями, используя один и тот же алгоритм, основанный на идеях рангового подхода [4, 5], суть которого состоит в следующем. Для определения свойств произвольной сложной системы можно использовать стянутое дерево всех путей графа, позволяющее с единых позиций решать произвольные задачи теории графов и комбинаторной оптимизации, в том числе задачи функционирования широкого круга систем управления. Использование стянутого дерева путей, в свою очередь, позволяет эффективно распараллелить процесс их решения на одном типе архитектур параллельных вычислительных систем (ПВС), что в некотором смысле снимает проблему, связанную с необходимостью разрабатывать различные архитектуры ПВС для различных типов задач.

Рассмотрим конечное множество произвольных объектов теории графов или конфигураций комбинаторной оптимизации. В общем случае объект может быть произвольным, но необходимо определить конечное множество элементов $\Omega = \{\omega_1\}$ или подмножества (ПМ) $L_i \in \Omega$ и правила R, позволяющие формировать объекты из исходных элементов, или ПМ L_i, принадлежащие множеству Ω.

Пусть задано некоторое разделение множества Ω на семейства ПМ $\{L_i\}$ такое, что $\bigcup_i L_i \in \Omega$, и L_i описывают объекты, состоящие из базовых элементов $\{l_i\}$ таких, что $\bigcup_i l_i \in \Omega$, и заданы правила R, позволяющие определять возможность объединения базовых элементов $\{l_i\}$, и правила P, определяющие весовые характеристики произвольных объединений $L_k \cup L_p \in \Omega$, характеризующие свойства $\{v\}$ рассматриваемых объектов. Определим возможность построения объекта с требуемым свойством $v^* \in \{v\}$.

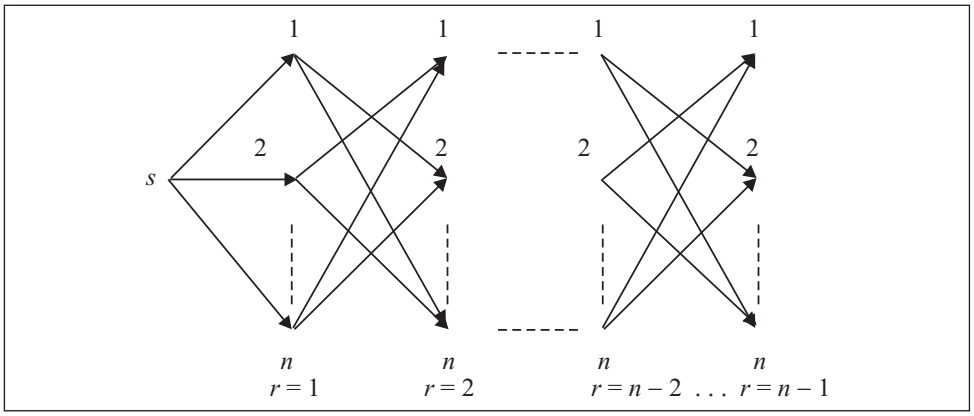

Рис. 1. Граф *D*

Представим множество всех возможных объединений ПМ L_i в виде графа *D* (рис. 1). Граф имеет параллельно ярусную структуру, состоит из *n* горизонтальных линеек с вершинами 1, 2, …, *n* и *n* ярусами, каждый из которых содержит все вершины графа *D*. Каждой вершине графа *D* поставим в соответствие базовый элемент l_i. Важной характеристикой пути в графе *D* является ранг *r* — число ребер, образующих путь. В графе *D* произвольная вершина *i* может быть достигнута путями рангов $r = 1$, $r = 2$, ..., $r = n - 1$, а произвольному пути μ_{st}, удовлетворяющему правилам построения *R* и проходящему через вершины (j, p, ..., k, t), соответствует объединение базовых элементов $(1_j \cup 1_p \cup ... \cup 1_k \cup 1_t)$ согласно правилам *R*, определяющим некоторый объект $L_i \in \Omega$. Длина этого пути $d(\mu_{st})$ определяется по правилам *P*.

Следовательно, множество всех путей $M_{sj}(r)$ в графе *D*, удовлетворяющих правилам *R*, определяет область допустимых решений исходной задачи по выделению объекта с интересующим нас свойством $v^* \in \{v\}$. В качестве исходной вершины в графе *D* используем фиктивную вершину *S*, которую в некоторых случаях удобно отождествлять с нулевым или исходным состоянием системы. Это приводит к тому, что максимальный ранг пути в графе *D* становится равным *n*, а добавление вершины *S* к базовым элементам системы не изменяет их свойств, определяемых правилами *R*.

Множество сложных систем, обладающих различными свойствами $\{F_i\}$, может быть отображено с помощью некоторого ПМ графов $\{G_i\}$. Рассмотрим произвольный граф $G(V, E) \in \{G_i\}$ с *n* вершинами, описывающий состояние системы $F \in \{F_i\}$ с конечным числом состояний *n*. Вершины $\{i\} \in V$ графа $G(V, E)$ соответствуют возможным состояниям системы. Пу-

ти в графе $G(V, E)$, определяемые последовательностью прохождения вершин $\{v_i\}$ и ребер $\{(i, j)\}$, характеризуют возможный порядок достижения состояния $i = p$ из некоторого исходного состояния s. В графе $G(V, E)$ максимальное значение ранга $r = n - 1$, и в общем случае ранг произвольного пути μ_{sp} характеризует сумму начального, конечного состояний и числа состояний предшествования, через которые может быть достигнуто состояние p из некоторого исходного состояния s. Тогда множества путей M_{sj}^r, $j = (\overline{1, n})$, определяют способы достижения состояния j.

Поставим в соответствие каждому базовому элементу из множества $\{l_i\}$ вершину графа $G(V, E)$. Тогда объектам $\{L_j\}$ будут соответствовать все множество объектов Ω, порождаемое на множестве V с использованием правил R и P, например клики в произвольных графах, независимые множества, циклы графов, векторы и матрицы, задающие некоторый объект в $G(V, E)$. Каждый объект будем характеризовать $m+1$ весовой характеристикой, где m — некоторые второстепенные характеристики объекта, на которые в общем случае могут быть наложены следующие ограничения:

они не должны превышать некоторых величин $\{b_i\}$, $i = (1, 2, ..., m)$;

должен быть один определяющий показатель качества объекта, построенного из исходных элементов или их подмножеств;

объект, принадлежащий множеству Ω, должен обладать определенным свойством v.

Таким образом, пути μ_{sj}^r в графе D соответствует объект L_j, который может быть построен из r базовых элементов, соответствующих вершинам $\{v_j\}$, включая элемент j на основе правил R. Множества путей M_{sj}^r, $j = (\overline{1, n})$, определяют множество объектов L_j, которые можно построить из r базовых элементов, соответствующих вершинам $\{v_j\}$, включая элемент j.

Формализация и постановка задачи. Для описания множества аддитивных целочисленных функций с произвольными нелинейностями, определяемыми на множестве переменных $\{X_1, X_2, ..., X_n\}$, введем порождающую функцию $F(x)$:

$$F(x) = \sum_{k=1}^{n} \sum_{j=1}^{p_k} C_{k,j} S_{k,j}, \qquad (1)$$

где $S_{k,j}$ — произведение k различных переменных; $C_{k,j}$ — целочисленные коэффициенты при $S_{k,j}$, $p_k = C_n^k = \dfrac{n!}{k!(n-k!)}$, $k = 1 \div n$. С ее помощью можно задавать и функционал и ограничения в задачах нелинейного булевого программирования.

Обозначим через H множество функций различных слагаемых в (1), которые можно получить, полагая равными нулю различные значения

$C_{k,j}$. Множество H является полным, т.е. содержит в себе все возможные нелинейности, состоящие из всех возможных сочетаний переменных, образующих эти нелинейности, которые можно построить на основе данного ПМ переменных $\{X_1, X_2, ..., X_n\}$. Нетрудно показать, что мощность данного множества очень велика, но конечна и равна 2^{p_Σ}, где

$$p_\Sigma = 1 + \frac{1}{2}\left[\begin{array}{l} \dfrac{n!}{1!(n-1)!}\left(\dfrac{n!}{1!(n-1)!}+1\right)+\dfrac{n!}{2!(n-2)!}\left(\dfrac{n!}{2!(n-2)!}+1\right)+ \\ +\dfrac{n!}{k!(n-k)!}\left(\dfrac{n!}{k!(n-k)!}+1\right)+\dfrac{n!}{(n-1)!1!}\left(\dfrac{n!}{(n-1)!1!}+1\right) \end{array}\right].$$

Следует заметить, что с помощью (1) можно определить класс задач нелинейного булевого программирования с положительными целочисленными коэффициентами, в которых решение определяется только сочетанием переменных и не зависит от их перестановки в $S_{k,j}$. В общем случае задачу булевого программирования можно представить в виде

$$f(X_1, X_2, ..., X_n) = \sum_{k=1}^{n} \sum_{j=1}^{p_k} C_{k,j} S_{k,j} \Rightarrow \max$$

при ограничениях

$$g_l(X_1, X_2, ..., X_n) = \sum_{k=1}^{n} \sum_{j=1}^{p_k} T_{k,j} S_{k,j} \le b_l, \ \ l = (\overline{1,m}), \qquad (2)$$

где $C_{k,j}$, $T_{k,j}$, b_l — положительные целые числа.

Рассмотрим полносвязный граф $G(X, E)$, вершинами которого являются переменные X_i. Выделим в графе G произвольную клику $Q = X_p X_r \ ... \ X_m$, состоящую из $r < n$ вершин, и рассмотрим включения $Q \subseteq S_{k,j} \in f(X_1, X_2, ..., X_n)$ и $Q \subseteq S_{k,j} \in g(X_1, X_2, ..., X_n)$. Каждое включение можно охарактеризовать суммами коэффициентов $C_{k,j}$, стоящими при $S_{k,j}$ в функционале $f(X_1, X_2, ..., X_n)$, и суммами коэффициентов $T_{k,j}$, стоящими при $S_{k,j}$ в ограничениях $g_j(X_1, X_2, ..., X_n)$. В общем случае произвольная клика Q всегда будет характеризоваться соответствующим весом по функционалу $f(X_1, X_2, ..., X_n)$ и не более чем m весами по ограничениям.

Решение задачи. Воспользуемся представлением исходного графа G (V, E) в виде симметричного дерева путей графа D (стянутого дерева путей) [6—8] и (см. рис. 1). Смысл такого представления заключается в следующем. Пусть все возможные состояния некоторой системы опреде-

ляются графом $G(V, E)$ с n вершинами, где вершины соответствуют возможным состояниям системы. Перейдем к пространству с $(n-1)^2$ состояниями. Для этого каждому из n состояний поставим в соответствие еще $(n-1)$ состояние, характеризующее способ достижения состояния из множества $\{1, 2, ..., n\}$. При этом в качестве добавляемых состояний определим ранг пути в графе $G(V, E)$, т.е. число ребер, образующих данный путь . Из вершины s графа $G(V, E)$ в произвольную вершину j можно попасть путем ранга $r = 1$, используя одно ребро, путем ранга $r = 2$, — используя два ребра, и т.д., путем ранга $r = n - 1$, — используя $n - 1$ ребро.

Такое пространство состояний можно представить в виде стянутого дерева путей D (см. рис. 1), которое содержит $(n - 1)$ горизонтальную линейку и $(n - 1)$ ярус. Для прочтения путей на каждой горизонтальной линейке можно быть только один раз. Исходя из стянутого дерева путей, для произвольной вершины j множество путей, ведущих в эту вершину из некоторой вершины s, можно представить в следующем виде: $m_s(j) = m_{sj}^{r=1} \cup m_{sj}^{r=2} \cup ... \cup m_{sj}^{r=n-1}$, $j = \overline{(1, n-1)}$, где $M_{sj}^r = \{\mu_{sj}^r\}$ — ПМ путей из произвольной вершины s в некоторую вершину j графа $G(V, E)$ ранга r.

Следует заметить, что дерево всех путей графа D может быть построено и от конкретной вершины i. В этом случае вершина $s = i$ и i-я горизонтальная линейка исключается из D. Например, при $i = 2$ дерево D будет иметь вид, представленный на рис. 2. Далее стянутое дерево путей будем использовать для построения однопроходных алгоритмов решения задачи (2), а дерево D, представленное на рис. 2, — для построения n-проходных алгоритмов.

Таким образом, используя граф D и вводя правила формирования путей следующего ранга, можно из произвольной вершины s поэтапно строить пути $\{\mu_{sj}^r\}$ произвольного ранга вплоть до ранга $r = n - 1$. В решаемой задаче под состоянием системы будем подразумевать различные способы объединения вершин графа D в клики. Тогда каждому пути $\{\mu_{sj}^r\}$ ранга r в графе D, проходящему через вершины $(v_h, v_k, ..., v_p)$ в исходном графе $G(V, E)$, соответствует клика из $r = 1$ вершины $(X_h X_k ... X_p)$, характеризующаяся соответствующим весом по функционалу $f(X_1, X_2, ..., X_n)$ и не более чем m весами по ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$. Весовые характеристики $\{d_{sj}^{r-1}\}$ произвольной клики $Q^{r-1}(j)$, состоящей из $r - 1$ вершины и определяемой одним из путей $\mu_{sj}^r \in M_{sj}^r$ ранга r в графе D, вычисляются по весам функционала суммированием коэффициентов ПМ $L_f = \{C_{k,j}\}$, стоящих при слагаемых $S_{k,j}$ в функционале, т.е. удовлетворяющих условию $Q \subseteq S_{k,j} \in f(X_1, X_2, ..., X_n)$. Аналогично определяются весовые характеристики по весам ограничений суммированием коэффициентов ПМ $L_B = \{T_{k,j}\}$, стоящих при слагаемых $S_{k,j}$ в ограничениях, т.е. удовлетво-

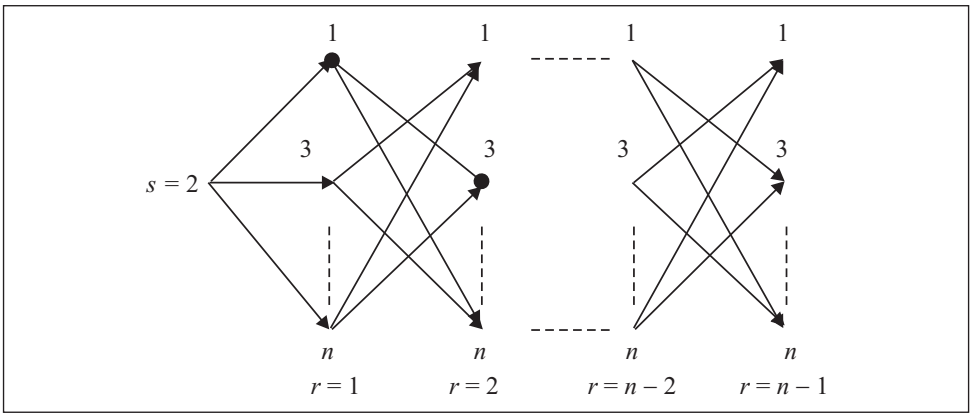

Рис. 2. Стянутое дерево всех путей графа D от вершины $s = 2$

ряющих условию $Q \subseteq S_{k,j} \in g(X_1, X_2, ..., X_n)$. Следовательно, весовые характеристики клик $Q^{r-1}(j)$ с множеством путей M_{sj}^r по весам функционала и ограничений определяются соответственно равенствами

$$d_{sj}^{f_{r-1}} = \sum_{C_{k,j} \in L_f} C_{k,j}, \quad d_{sj}^{B_{r-1}} = \sum_{T_{k,j} \in L_B} T_{k,j}. \tag{3}$$

Итак, в графе D каждый путь имеет в общем случае $m + 1$ длину: одну по весам функционала и m по весам ограничений. Для решения поставленной задачи в графе D нужно построить путь максимальной длины по весам функционала от вершин $1, 2, ..., n$ ко всем остальным вершинам графа, и при этом его длины по весам ограничений не должны превышать соответствующей величины b_j. Если на основе ПМ путей $M_{sj}^{r=1}$ в графе D строить ПМ $M_{sj}^{r=2}$ и так далее, до путей $M_{sj}^{r=n-1}$ ранга $r = n - 1$, то возникает необходимость построить $(n-1)!$ путей. Поэтому для формирования путей вводится процедура A, позволяющая отсекать неперспективные пути. Для отсечения неперспективных вариантов в процедуре A предлагается использовать принцип оптимизации по направлению к произвольной вершине p при формировании путей следующего ранга M_{sp}^{r+1} на основе путей предыдущего ранга M_{sj}^r [6—8]. Для рассматриваемой задачи это определяется следующим рекуррентным соотношением:

$$\mu_{sp}^{r+1} = \max_j \{\{\mu_{sp}^r\} \cup (j, p)\}, \quad j = (\overline{1, n}), \quad p = (\overline{1, n}), \quad j \neq p, \tag{4}$$

где (j, p) — ребро графа D; n — число различных вершин в графе D.

Рассмотрим возможность построения n-проходных и однопроходных процедур решения задачи (2) соответственно на стянутых деревьях

(см рис. 1 и 2). Перед началом работы процедуры A_1 переменной i присвоим значение 1 (i: = 1).

Процедура A_1 с n проходами. Ш а г 1. $s := i$. Из вершины s строим все возможные пути ранга $r = 1$ ко всем вершинам графа D (см. рис. 2), удовлетворяющие ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$. Длины по весам функционала и ограничениям вычисляем в соответствии с (3).

Ш а г 2. Используя пути текущего ранга r, строим все возможные пути ранга $r := r + 1$, удовлетворяющие ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$, с использованием рекуррентного соотношения (3). Проверка ограничений и выбор пути максимального по весам функционала осуществляется на основе вычислений длин путей по весам функционала и ограничений в соответствии с (3). Если в процессе применения рекуррентного соотношения (4) возникают несколько путей одинаковой длины, то выбираем любой из них.

Ш а г 3. Проверяем равенство $M_{sj}^{r+1} = \varnothing$. Если оно выполняется, то путь μ_{sj}^{*r} максимальной длины, полученный на ранге r, является локальным экстремумом решаемой задачи, иначе — выполняем следующий шаг.

Ш а г 4. Проверяем равество $i = n - 1$. Если оно не выполняется, то $i := i + 1$ и переходим к выполнению шага 1, иначе процедура A_1 заканчивает работу. При этом из множества локальных экстремумов $\{\mu_{sj}^{*r}\}$ выбираем глобальный экстремум μ_{sj}^{**r}, соответствующий оптимальному решению задачи (2).

Для уменьшения временной сложности работы алгоритма можно использовать однопроходный вариант реализации данной процедуры на основе стянутого дерева путей (см. рис. 1). При этом однопроходная процедура A_2 имеет следующий вид.

Процедура A_2 с одним проходом. Ш а г 1. Из вершины s строим все возможные пути ранга $r = 1$ ко всем вершинам графа D (см. рис. 2), удовлетворяющие ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$. Длины по весам функционала и ограничениям вычисляются в соответствии с (3).

Ш а г 2. Используя пути текущего ранга r, строим все возможные пути ранга $r := r + 1$, удовлетворяющие ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$, с использованием рекуррентного соотношения (4). Проверка ограничений и выбор пути максимального по весам функционала осуществляется на основе вычислений длин путей по весам функционала и ограничений в соответствии с соотношениями (3). Если в процессе применения (4) возникают несколько путей одинаковой длины, то выбираем любой из них.

Ш а г 3. Проверяем равенство $M_{sj}^{r+1} = \varnothing$. Если оно выполняется, то путь μ_{sj}^{*r} максимальной длины, полученный на ранге r, является локальным экстремумом решаемой задачи, иначе — выполняем следующий шаг.

Ш а г 4. Проверяем равенство $r = n$. Если оно не выполняется, то переходим к выполнению шага 2, иначе процедура A_2 заканчивает работу, и путь μ_{sj}^{*r} максимальной длины, полученный на ранге $r = n$, соответствует оптимальному решению задачи (2).

Еще одним вариантом уменьшения временной сложности алгоритмов на основе процедур A_1 и A_2 могут быть процедуры A' и A'', отличающиеся от A_1 и A_2 тем, что на каждом ярусе формирования путей на основе процедур A_1 и A_2 локальные экстремумы будут выделяться не в каждом множестве. В этом случае выделяются глобальные экстремумы на ярусе и на основе пути, соответствующего глобальному экстремуму, формируются пути следующего яруса, удовлетворяющие ограничениям. При этом процедуры A' и A'' будут иметь следующий вид.

Процедура A′. Перед началом работы процедуры A' переменной i присваиваем значение 1.

Ш а г 1. $s := i$. Из вершины s строим все возможные пути ранга $r = 1$ ко всем вершинам графа D (см. рис. 2), удовлетворяющие ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$. Длины по весам функционала и ограничениям вычисляем в соответствии с соотношениями (3). Затем выделяем самый длинный путь на ярусе.

Ш а г 2. Используя самый длинный путь текущего ранга r, построенный на предыдущем шаге, строим все возможные пути ранга $r := r + 1$, удовлетворяющие ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$, с использованием рекуррентного соотношения (4). Проверка ограничений и выбор пути максимального по весам функционала осуществляется на основе вычислений длин путей по весам функционала и ограничений в соответствии с соотношениями (3).

Ш а г 3. Проверяем равество $M_{sj}^{r+1} = \varnothing$. Если оно выполняется, то путь μ_{sj}^{*r} максимальной длины, полученный на ранге r, является локальным экстремумом решаемой задачи, иначе — выполняем следующий шаг.

Ш а г 4. Проверяем равенство $i = n - 1$. Если оно не выполняется, то $i := i + 1$, и переходим к выполнению шага 1, иначе процедура A' заканчивает работу. При этом из множества локальных экстремумов, полученных за один проход, из множества $\{\mu_{sj}^{*r}\}$ выбирается глобальный путь μ_{sj}^{**r}, соответствующий оптимальному решению задачи (2).

Процедура A″. Ш а г 1. Из вершины s строим все возможные пути ранга $r = 1$ ко всем вершинам графа D (см. рис. 1), удовлетворяющие ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$. Длины по весам функционала и ограничениям вычисляются в соответствии с соотношениями (3). Затем выделяем самый длинный путь на ярусе.

Ш а г 2. Используя самый длинный путь текущего ранга r, построенный на предыдущем шаге, строим все возможные пути ранга $r := r + 1$,

удовлетворяющие ограничениям $g_l(X_1, X_2, ..., X_n)$, $l = \overline{(1, m)}$, с использованием рекуррентного соотношения (4). Проверка ограничений и выбор пути максимального по весам функционала осуществляется на основе вычислений длин путей по весам функционала и ограничений в соответствии с соотношениями (3).

Ш а г 3. Проверяем равенство $M_{sj}^{r+1} = \varnothing$. Если оно выполняется, то путь μ_{sj}^{*r} максимальной длины, полученный на ранге r, является локальным экстремумом решаемой задачи, иначе — выполняем следующий шаг.

Ш а г 4. Проверяем равенство $r = n$. Если оно не выполняется, то переходим к выполнению шага 2, иначе процедура A'' заканчивает работу, и путь μ_{sj}^{*r} максимальной длины, полученный на ранге $r = n$, соответствует оптимальному решению задачи (2).

Обозначим через A_5 алгоритм, реализуемый на основе многопроходной процедуры, через A_4 — алгоритм, реализуемый на основе однопроходной процедуры A_2, и через A_3 — алгоритм, реализуемый на основе однопроходной процедуры A'', охватывающей все способы выделения локальных экстремумов.

Оценка сложности процедур {*A*}. При работе процедур $\{A\}$ число путей на произвольном ранге r не может превысить величины $(n-1)(n-1)$. Максимальный ранг r произвольного пути не превышает $(n-1)$ и число циклов, выполняемое процедурой A_1, равно n. Следовательно, после n циклов число путей, построенное процедурой A_1, не может превысить $(n-1)(n-1)(n-1)\,n \approx n^4$, а число обработанных векторов не может превысить n^5. С учетом числа слагаемых k в функционале и числа ограничений m временная сложность алгоритма не превысит в худшем случае $O(n^5 k(m+1))$. В случае, когда решение задачи осуществляется за один проход процедуры A_2 или за один проход процедуры A'', но с выделением наиболее длинного пути на ярусе, сложность процедур A_2 и A'' не превысят соответственно $O(n^4 k(m+1))$ и $O(n^3 k(m+1))$. Итак, алгоритмы A_5, A_4, A_3 имеют временную сложность, не превышающую в худшем случае соответственно $O(n^5 k(m+1))$, $O(n^4 k(m+1))$ и $O(n^3 k(m+1))$.

Таким образом, предложенный подход решения произвольных задач булевого программирования позволяет на основе разработанных алгоритмов решать с единых позиций любые задачи линейного и нелинейного булевого программирования за полиномиальное время с требуемой точностью.

Экспериментальное исследование алгоритмов на основе разработанных процедур. В качестве основных характеристик приближенных алгоритмов используем число элементарных операций и число обработанных векторов, функцию временных затрат и относительную погреш-

ность $П$ полученного приближенного решения $П = \frac{|f(x) - f(x^*)|}{f(x^*)}$, где f — целевая функция, определенная на некотором множестве M; x — приближенное решение задачи; x^* — оптимальное решение.

Были исследованы алгоритм A_5 на основе многопроходной процедуры A_1, алгоритм A_4 на основе однопроходной процедуры A_2 и алгоритм A_3 на основе однопроходной процедуры A''. При исследовании коэффициенты в функционале и ограничениях генерировались по равномерному закону распределения в функционале в диапазоне от 0 до 10, а в ограничениях — от 0 до 20. На каждую точку при оценке временной сложности алгоритмов в среднем и их погрешности решалось не менее 50 тестовых задач, результаты которых получены с доверительной вероятностью 0,95.

В качестве точного использован алгоритм решения задач нелинейного и линейного программирования, разработанный на основе идей рангового подхода для прогнозирования перспективных решений. Для отсева неперспективных путей использован метод ветвей и границ, позволивший устранить погрешности, не превышающие размерности $n = 70$.

Графики зависимости погрешности от размерности решаемых задач и от числа ограничений в задаче (2) приведены на рис. 3 и 4, из которых видно, что погрешность алгоритмов с увеличением числа ограничений m асимптотически уменьшается, а с увеличением n возрастает, при этом $m \geq 50$. Погрешность алгоритмов стабилизируется и для задач линейного программирования не превышает 2 %, а для задач квадратичного программирования не превышает 5—10 %. Решение тестовых задач показало, что увеличение диапазона изменения коэффициентов в функционале и ограничениях приводит к резкому снижению погрешностей алгоритмов. Переход от нелинейностей одного порядка к нелинейностям более высокого порядка может приводить к незначительному возрастанию погрешностей при небольшом числе ограничений, а с возрастанием числа ограничений возрастание погрешности очень быстро компенсируется.

Результаты экспериментального исследования временной сложности свидетельствуют о том, что число обрабатываемых векторов не зависит от числа ограничений и в среднем для алгоритмов A_5, A_4, A_3 временная сложность составляет соответственно $O(0{,}1n^{4,9})$, $O(0{,}3n^{3,7})$ и $O(0{,}4n^{2,8})$.

Пример. Решим задачу:

$$\begin{gathered} f(x) = 20x_1x_2 + 13x_1x_4 + 17x_3x_4 + 20x_3 + 10x_4 \to \max, \\ 2x_1x_2 + 10x_3x_4 + 7x_1x_4 + 4x_4 + 3x_2x_4 \leq 10. \end{gathered} \tag{5}$$

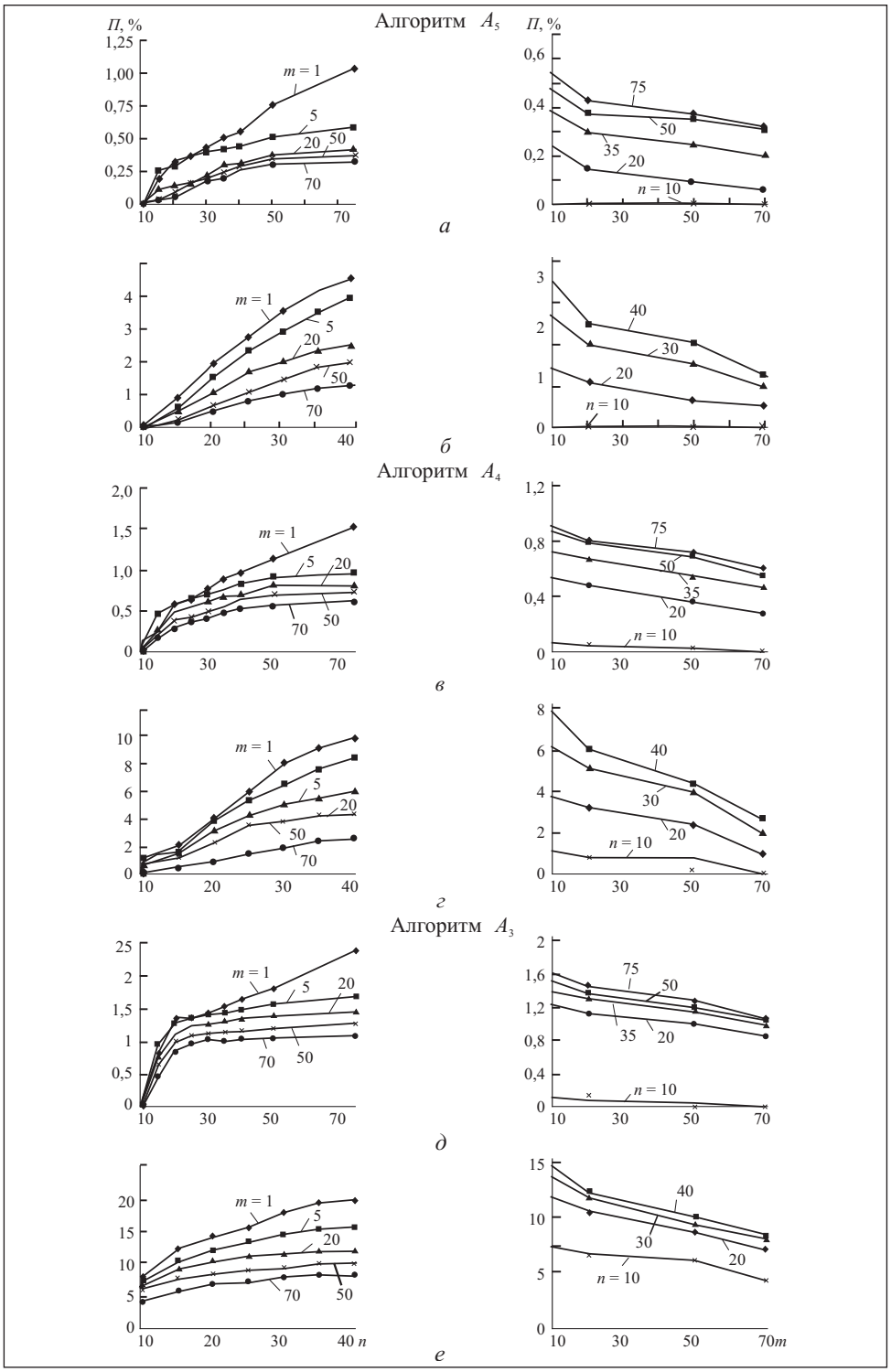

Рис. 3. Зависимость погрешности алгоритмов A_5 (*а*, *б*), A_4 (*в*, *г*) и A_3 (*д*, *е*) от размерности n решаемой задачи линейного (*а*, *в*, *д*) и квадратичного (*б*, *г*, *е*) булевого программирования при различном числе ограничений m

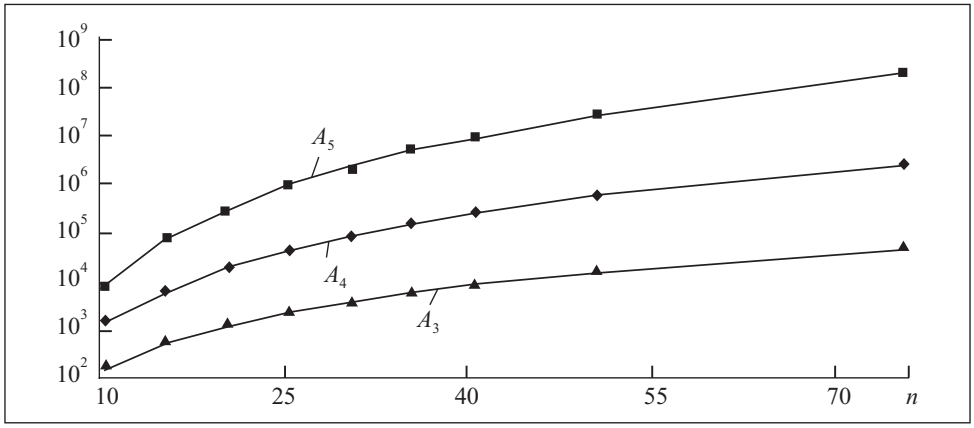

Рис. 4. Зависимость числа обрабатываемых векторов от размерности решаемой задачи линейного и квадратичного программирования для алгоритмов A_5, A_4, A_3

Пространство возможных решений представим в виде графа D. Процесс формирования путей в графе, где каждой его вершине соответствует номер i, имеет следующий вид:

$S1(0,0)$	1	$S21(20,2)$ $S31(20,0)$	1	$S321(40,2)$ $S231(40,2)$	1
$S2(0,0)$	2	$S12(20,2)$ $S32(20,0)$ $S42(10,7)$	2	$S312(40,2)$ $S132(40,2)$	2
$S3(20,0)$	3	$S13(20,0)$ S23(20,0)	3	$S213(40,2)$ $S123(40,2)$	3
$S4(10,4)$	4	$S24(10,7)$	4	$\varnothing$	4

Формируем сначала все возможные пути из вершины S к вершинам 1, 2, 3, 4. Это пути $S1$, $S2$, $S3$, $S4$. Каждый путь имеет две длины: длина по весам коэффициентов в функционале (первая в скобках) и длина по весам коэффициентов в ограничении (вторая в скобках).

Определяем длины пути $S1$. Проверяем, присутствует в функционале отдельно переменная x_1 или нет. Она не присутствует отдельно ни в функционале, ни в ограничениях, поэтому $S1(0,0)$ и аналогично — $S2(0,0)$. Далее, для пути $S3$ проверяем, присутствует ли отдельно в функционале переменная x_3. Да, при ней стоит коэффициент 20, значит, длина пути по функционалу равна 20. В ограничении отдельно переменная x_3 не присутствует, следовательно, длина пути по весам ограничений равна 0, т. е. путь $S3$ имеет две длины (20, 0).

Для пути $S4$ проверяем, присутствует ли в функционале отдельно переменная x_4. Да, при ней стоит коэффициент 10, значит, длина пути по функционалу равна 10. Проверяем, в ограничении переменная x_4 отдельно присутствует? Да, при ней стоит коэффициент 4, значит, длина пути по ограничению равна 4, т.е. путь $S4$ имеет две длины (10, 4). Далее на основе сформированных путей $S1$, $S2$, $S3$, $S4$ формируем все возможные пути следующего ранга к вершинам 1, 2, 3, 4, длины которых по весам ограничений не превышают 10. Так, из вершины 1 можно попасть в вершины 2, 3 и 4.

Формируем путь $S12$ и определяем его длины. Проверяем, присутствуют или нет в функционале переменные x_1 и x_2 отдельно или совместно. Да, присутствуют, при произведении $x_1\,x_2$ стоит коэффициент 20, значит, путь $S12$ по весам функционала имеет вес 20. Проверяем, присутствуют или нет в ограничениях переменная x_1 и x_2 отдельно или совместно. Да, в ограничениях переменные x_1 и x_2 присутствуют в виде произведения x_1x_2, при котором стоит коэффициент 2. Значит, путь $S12$ по весам ограничений имеет вес 4, т.е. путь $S12$ имеет две длины (20, 2).

Формируем путь $S13$ и определяем его длины. Проверяем, присутствуют или нет в функционале переменные x_1 и x_3 отдельно или совместно. Да, при переменной x_3 стоит коэффициент 20, а совместно переменные x_1 и x_3 в функционале не присутствуют. Значит, путь $S13$ по весам функционала имеет вес, равный 20. Проверяем, присутствуют или нет в ограничениях переменная x_1 и x_3 отдельно или совместно. Нет, следовательно, длина пути $S13$ по весам ограничений равна 0, т.е. путь $S13$ имеет две длины (20, 0).

Формируем путь $S14$ и определяем его длины. Проверяем, присутствуют или нет в функционале переменные x_1 и x_4 отдельно или совместно. Да, при переменной x_4, стоящей отдельно, стоит коэффициент 10, а при произведении x_1x_4 в функционале стоит коэффициент 13. Значит, путь $S14$ по весам функционала имеет вес, равный 10 + 13 = 23. Проверяем, присутствуют или нет в ограничениях переменные x_1 и x_4 отдельно или совместно. Да, присутствуют, при переменной x_4 стоит коэффициент 4, а при произведении x_1x_4 в ограничениях стоит коэффициент 7. Значит, путь $S14$ по весам имеет вес, равный 7 + 4 = 11. Следовательно, длина пути $S14$ по весам ограничений равна 11, т.е. путь $S14$ имеет две длины (23, 11), но поскольку длина по весам ограничений превышает правую часть ограничения, этот путь исключаем из дальнейшего анализа. Поэтому в множество 4 на втором ярусе этот путь не попал.

Аналогично формируем все возможные пути от вершин 2, 3 и 4.

Далее, на основе путей, сформированных на втором ярусе, строим все возможные пути следующего ранга. При этом будем использовать прин-

цип оптимизации по направлению, который заключается в том, что из каждого множества путей *i* формируем во все оставшиеся ПМ пути максимально возможного веса по функционалу, длина которых по весам ограничений не превышает правой части ограничения.

Начнем с множества 1, которое содержит два пути: *S*21 (20, 2) и *S*31 (20, 0). Из множества 1 можно попасть в множества 2, 3 и 4. Формируем все возможные пути из множества 1 в 2, используя принцип оптимизации по направлению: путь *S*21 (20, 2) в множество 2 не может быть продлен, так как вершина 2 уже вошла в путь *S*21 (20, 2) . Следовательно, выбираем один оставшийся путь *S*31 (20, 0) . В результате получаем путь *S*312, длина которого определяется так: проверяем, присутствуют или нет в функционале переменные x_1, x_2 и x_3 отдельно и в комбинациях $x_1 x_2, x_1 x_3, x_2 x_3$ или в виде сомножителя $x_1 x_2 x_3$, т.е. определяем, есть ли произведения переменных, которые входят в сомножитель $x_1 x_2 x_3$, и коэффициенты при этих сомножителях суммируем. Да, есть переменная x_3 с коэффициентом 20, при этом есть сомножитель $x_1 x_2$ с коэффициентом 20. Следовательно, длина пути *S*312 по весам функционала будет 20 + 20 = 40.

Аналогично определяется длина пути по весам ограничения. В ограничениях есть сомножители, входящие в произведение $x_1 x_2 x_3$, это только $x_1 x_2$ с коэффициентом 2 при нем. Следовательно, длина пути *S*312 по весам ограничений равна 2, т.е. данный путь *S*312 имеет две длины (40, 2). Если во множестве 1 остался не один путь *S*31(20, 0), а несколько путей, которые могут пройти по ограничениям в множество 2, то из них в 2 перетягиваем только один путь с максимальным весом по функционалу.

Формируем все возможные пути из множества 1 в 3, используя принцип оптимизации по направлению. Путь *S*31(20, 0) в множество 3 не может быть продлен, так как третья вершина уже вошла в путь *S*31(20, 0). Следовательно, выбираем один оставшийся путь *S*21(20, 0). В результате получаем путь *S*213, длину которого определяем так. Проверяем, присутствуют или нет в функционале переменные x_1, x_2 и x_3 отдельно и в комбинациях $x_1 x_2, x_1 x_3, x_2 x_3$ или в виде сомножителя $x_1 x_2 x_3$, т.е. смотрим, есть ли произведения переменных, которые входят в сомножитель $x_1 x_2 x_3$, и коэффициенты при этих сомножителях суммируем. Да, есть переменная x_3 с коэффициентом 20 и есть сомножитель $x_1 x_2$ с коэффициентом 20. Следовательно, длина пути *S*213 по весам функционала будет 20 + 20 = 40.

Аналогично определяется длина пути по весам ограничения. В ограничениях есть сомножители, входящие в произведение $x_1 x_2 x_3$. Это только $x_1 x_2$ с коэффициентом 2 при нем. Значит, длина пути *S*213 по весам ограничений равна 2, т.е. данный путь *S*213 имеет две длины (40,2).

Далее формируем все возможные пути из множества 1 в 4, используя принцип оптимизации по направлению. Легко проверить, что ни один из

путей, которые можно построить в множество 4, не удовлетворяют ограничению, и их можно исключить из анализа. Аналогично формируются пути из множества 2 в множества 1, 3, 4 и пути из множества 3 в множества 1, 2, 4, а также пути из множества 4 в множества 1, 2, 3. При этом последние не пройдут по ограничениям ни в одно множество.

На следующем ранге не удастся построить ни одного пути, так как все они не удовлетворяют ограничению. Таким образом, самым длинным путем является путь $S213$. Он имеет две длины (40, 2) и, следовательно, вектор, обращающий в максимум функционал (5), равный 40, имеет вид $X = \{x_1 = 1, x_2 = 1, x_3 = 1, x_4 = 0\}$. При этом значение ограничения не превышает 2. Следует заметить, что пути, содержащие одинаковые ПМ вершин, имеют одинаковые длины по весам функционала и ограничениям. Поэтому они дублируют друг друга на ярусе и их можно исключать. В этом случае процесс построения путей будет иметь следующий вид:

$S1(0,0)$	1	$S31(20,0)$	1		1
$S2(0,0)$		$S12(20,2)$ $S32(20,0)$		$S312(40,2)$	
	2	$S42(10,7)$	2		2
$S3(20,0)$	3		3		3
$S4(10,4)$	4		4		4

В случае, когда имеется не одно, а m ограничений, каждый путь характеризуется не двумя длинами, а $m+1$ длиной, определяемой аналогично.

Рассмотренные процедуры решения задач булевого программирования будем называть первым типом оптимизации. Для повышения эффективности работы алгоритма введем второй тип оптимизации, используя аналогичную процедуру формирования путей, которая отличается от процедуры первого типа тем, что на первом ярусе будем формировать не пути ранга 1, а множества путей, соответствующие слагаемым в функционале. Решение задачи (5) в этом случае имеет следующий вид:

$S21(20,2)$	1	$S31(20,0)$	1		1
		$S32(20,0)$		$S312(40,2)$	
	2	$S42(10,7)$	2		2
$S3(20,0)$	3	$S213(20,0)$	3		3
$S4(10,4)$	4		4		4

Пути $S41(33,11)$ и $S43(47,14)$ на первом ярусе не формировались, так как они не проходят по ограничению. Сравнивая процессы формирования путей в графе, видим, что число путей при использовании оптимизации

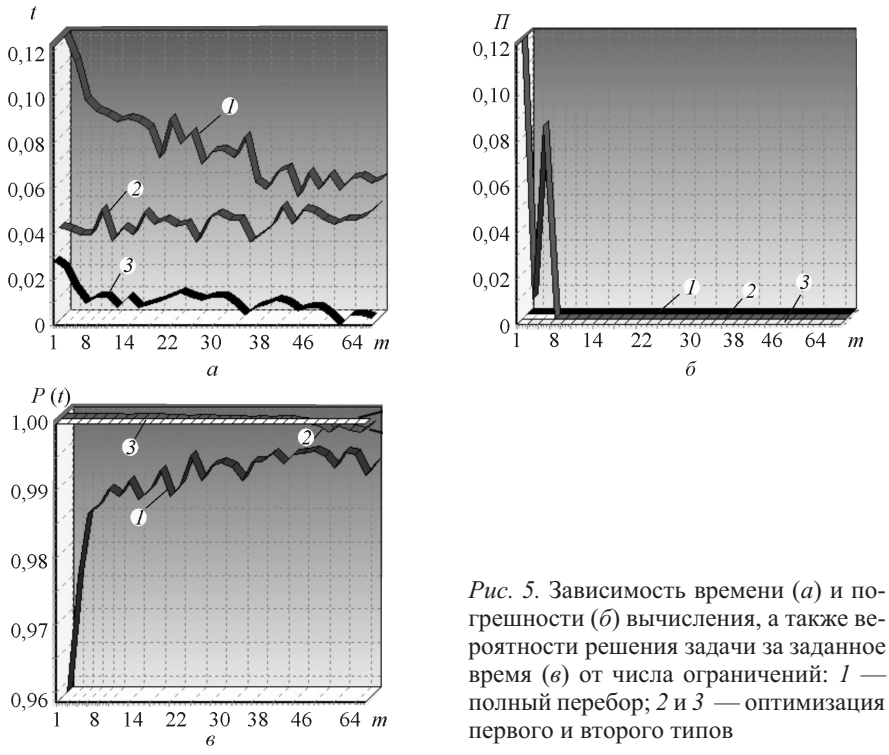

Рис. 5. Зависимость времени (*а*) и погрешности (*б*) вычисления, а также вероятности решения задачи за заданное время (*в*) от числа ограничений: *1* — полный перебор; *2* и *3* — оптимизация первого и второго типов

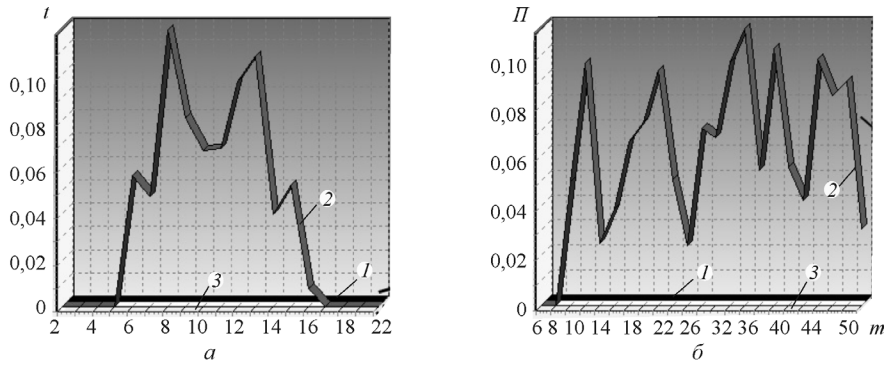

Рис. 6. Зависимость погрешности вычисления от числа переменных (*а*) и числа слагаемых (*б*): *1* — полный перебор; *2* и *3* — оптимизация первого и второго типов

второго типа меньше, в связи с чем уменьшается временная сложность предлагаемой процедуры. Поэтому было выполнено экспериментальное исследование и проведен сравнительный анализ двух типов оптимизации для оценки их временной сложности и погрешности. Значение показателя

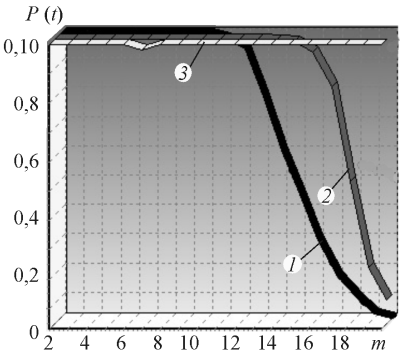

Рис. 7. Зависимость вероятности решения задачи от числа переменных: *1* — полный перебор; *2* и *3* — оптимизация первого и второго типов

оперативности их решения оценивается вероятностью решения задачи $P(T) = 1 - e^{-(T_\partial / T)}$ за некоторое допустимое время T_∂. В качестве эталонного точного алгоритма применен алгоритм полного перебора с использованием стратегии ветвей и границ. При проведении эксперимента число ограничений изменялось от 7 до 68, число переменных — от 2 до 20, число слагаемых в функционале и ограничениях — от 2 до 50. Диапазон изменения коэффициентов в функционале и ограничениях изменялся от 0 до 200 по равномерному закону распределения. Время решения приведено в секундах, допустимое время T_∂ = 30 с. При оценке временной сложности алгоритмов и их погрешности для каждой точки в среднем решалось не менее 50 тестовых задач.

Результаты экспериментального исследования временной сложности и погрешности разработанных алгоритмов подтверждают их теоретическую оценку (рис. 5—7).

Выводы

Использование предложенных процедур оптимизации второго типа позволяет значительно повысить оперативность решения задач булевого программирования и точность их решения, что имеет существенное значение при планировании выполнения заданий в распределенных вычислительных системах. Предложенные процедуры могут быть эффективно распараллелены, так как формирование путей на ярусах в множествах M^r_{sj} можно осуществлять одновременно. Поэтому при наличии n процессорных элементов для формирования путей временная сложность алгоритмов на основе рассматриваемых процедур A_1 и A_2 не превысит соответственно $O(pn^2)$ и $O(pn)$. Использование CUDA технологий позволит применять данные процедуры в масштабе реального времени в распределенных системах вычисления.

СПИСОК ЛИТЕРАТУРЫ

1. *Пападимитриу Х., Стайглиц К.* Комбинаторная оптимизация. Алгоритмы и сложность. — М. : «Мир», 1985. — 509 с.
2. *Foster C., Kesselman S.T.* The Anatomy of the Grid: Enabling Scalable Virtual Organizations // Intern. J. Supercomputer Applications. — 2001. — 15 (3). [Электронный ресурс]. — Режим доступа: http://www.globus.org/alliance/publications/papers/anatomy.pdf
3. *Пономаренко В.С., Листровой С.В., Минухин С.В., Знахур С.В* Методы и модели планирования ресурсов в GRID-системах. — Харьков: ИД «ИНЖЭК», 2008. — 408 с.
4. *Листровой С.В., Минухин С.В.* Общий подход к решению задач оптимизации в распределенных вычислительных системах и теории построения интеллектуальных систем // Проблемы управления и информатика. — 2010. — № 2. — С. 65—82.
5. *Listrovoy S.V., Minukhin S.V.* General Approach to Solving Optimization Problems in Distributed Computing Systems and Theory of Intelligence Systems Construction // Journal of automation and information sciences. — 2010. — V. 42, N 3. — P. 30—46.
6. *Listrovoy S.V., Golubnichiy D.Yu., Listrovaya E.S.* Solution method on the basis of rank approach for integer linear problems with boolean variables// Engineering Simulation. — 1999. — V. 16. — P. 707—725.
7. *Listrovoy S.V., Tretjak V.F., Listrovaya A.S.* Parallel algorithms of calculation process optimization for the boolean programming problems// Ibid. — 1999. — V. 16. — P. 569—579.
8. *Listrovoy S.V.* A.Yu. GUL Method of Minimum Covering Problem Solution on the Basis of Rank Approach// Engineering Simulation. — 1999. — V. 17. — P. 73—89.

REFERENCES

1. Papadimitriu, H. and Stayglits, K. (1985), *Kombinatornaya optimizatsiya. Algoritmy i slozhnost* [Combinatorial optimization. Algorithms and complexity], Mir, Moscow, Russia.
2. Foster, C. and Kesselman, S. (2001), The anatomy of the Grid: Enabling scalable virtual organizations, *Intern. J. Supercomputer Applications*, Vol. 15, no. 3, available at: http://www.globus.org/alliance/publications/papers/anatomy.pdf.
3. Ponomarenko, V.S., Listrovoy, S.V. and Minukhin, S.V. (2008), *Metody i modeli planirovaniya resursov v GRID-sistemakh. Monografiya* [Methods and models of scheduling resources in GRID-systems. Monograph], Izdatelskii dom INZhEK, Kharkov, Ukraine.
4. Listrovoy, S.V. and Minukhin, S.V. (2010), "General approach to solutions of optimization problems in distributed computer systems and theory of intelligence systems construction", *Problemy upravleniya i informatika*, no. 2, pp. 65-82.
5. Listrovoy, S.V. and Minukhin S.V. (2010), "General approach to solving optimization problems in distributed computing systems and theory of intelligence systems construction", *Journal of automation and information sciences*, Vol. 42, no. 3, pp. 30-46.
6. Listrovoy, S.V. , Golubnichiy, D.Yu. and Listrovaya, E.S. (1999), "Solution method on the basis of rank approach for integer linear problems with Boolean variables", *Engineering Simulation,* Vol. 16, pp. 707-725.
7. Listrovoy, S.V., Tretjak, V.F. and Listrovaya, A.S. (1999), "Parallel algorithms of calculation process optimization for the Boolean programming problems", *Ibid*., Vol. 16, pp. 569-579.
8. Listrovoy, S.V. (1999), "A.Yu. GUL method of minimum covering problem solution on the basis of rank approach", *Engineering Simulation*, Vol. 17, pp. 73-89.

S.V. Listrovoy, E.S. Listrovaya, M.S. Kurtsev

RANK APPROACH TO THE SOLUTION OF PROBLEMS OF LINEAR AND NONLINEAR BOOLEAN PROGRAMMING FOR PLANNING AND MANAGEMENT IN DISTRIBUTED COMPUTING SYSTEMS

The efficiency of ranking approach to solving arbitrary Boolean programming tasks has been shown. Procedures are described which allow solving problems of linear and nonlinear programming using algorithms of polynomial complexity with a small error, with arbitrary nonlinearities, both in functionality and limitations. The article also shows the results of experimental investigation of the error of the developed algorithms and their time complexity.

K e y w o r d s: discrete programming, ranking approach, planning, distributed computing system.

Поступила 04.05.16;
после доработки 09.12.16

ЛИСТРОВОЙ Сергей Владимирович, д-р техн. наук, профессор Украинского государственного университета железнодорожного транспорта (г. Харьков). В 1972 г. окончил Харьковское высшее военное командно-инженерное училище. Область научных исследований — задачи дискретной оптимизации и теории графов и их приложения к анализу вычислительных систем и сетей.

ЛИСТРОВАЯ Елена Сергеевна, канд. техн. наук, доцент кафедры экономики и маркетинга Национального аэрокосмического университета им. Н.Е. Жуковского (г. Харьков), который окончила в 1998 г. Область научных исследований — применение информационных систем в экономической сфере деятельности.

КУРЦЕВ Максим Сергеевич, аспирант Украинского государственного университета железнодорожного транспорта (г. Харьков). В 2010 г. окончил Украинскую государственную академию железнодорожного транспорта. Область научных исследований — задачи управления и планирования в распределенных вычислительных системах.